**Exception Handling – Part-08 – Customized Exception**

* **Exception handling keywords summary:**

try 🡪 To maintain risky code.

catch 🡪 To maintain exception handling code.

finally 🡪 To maintain cleanup code.

throw 🡪 To hand-over our own exception object to the JVM

manually.

throws 🡪 To delegate responsibility of exception handling to the

caller.

* **Various possible compile time errors in exception handling:**

1. Unreported exception XXX; must be caught or declared to be thrown.
2. Exception XXX has already been caught.
3. Exception XXX is never thrown in body of corresponding try statement.
4. Unreachable statement.
5. Incompatible types:

found: Test

required: java.lang.Throwable

1. Try without catch or finally.
2. catch without try.
3. finally without try.

* **Customized or user defined exceptions:**

Sometimes to meet programming requirements we can define our own exceptions such type of exceptions are called customized or user-defined exceptions.

Example:

class TooYoungException extends RuntimeException{

TooYoungException(String message){

super(message);

}

}

// super – To make description available to default exception handler.

class TooOldException extends RuntimeException{

TooOldException(String message){

super(message);

}

}

class CustException{

public static void main(String[] args){

int age = Integer.parseInt(args[0]);

if(age > 60){

throw new TooYoungException(“Please wait somemore time… You will get best match soon”);

} else if (age < 18){

throw new TooOldeException(“Your age already crossed marriage age… No chance of getting marriage”);

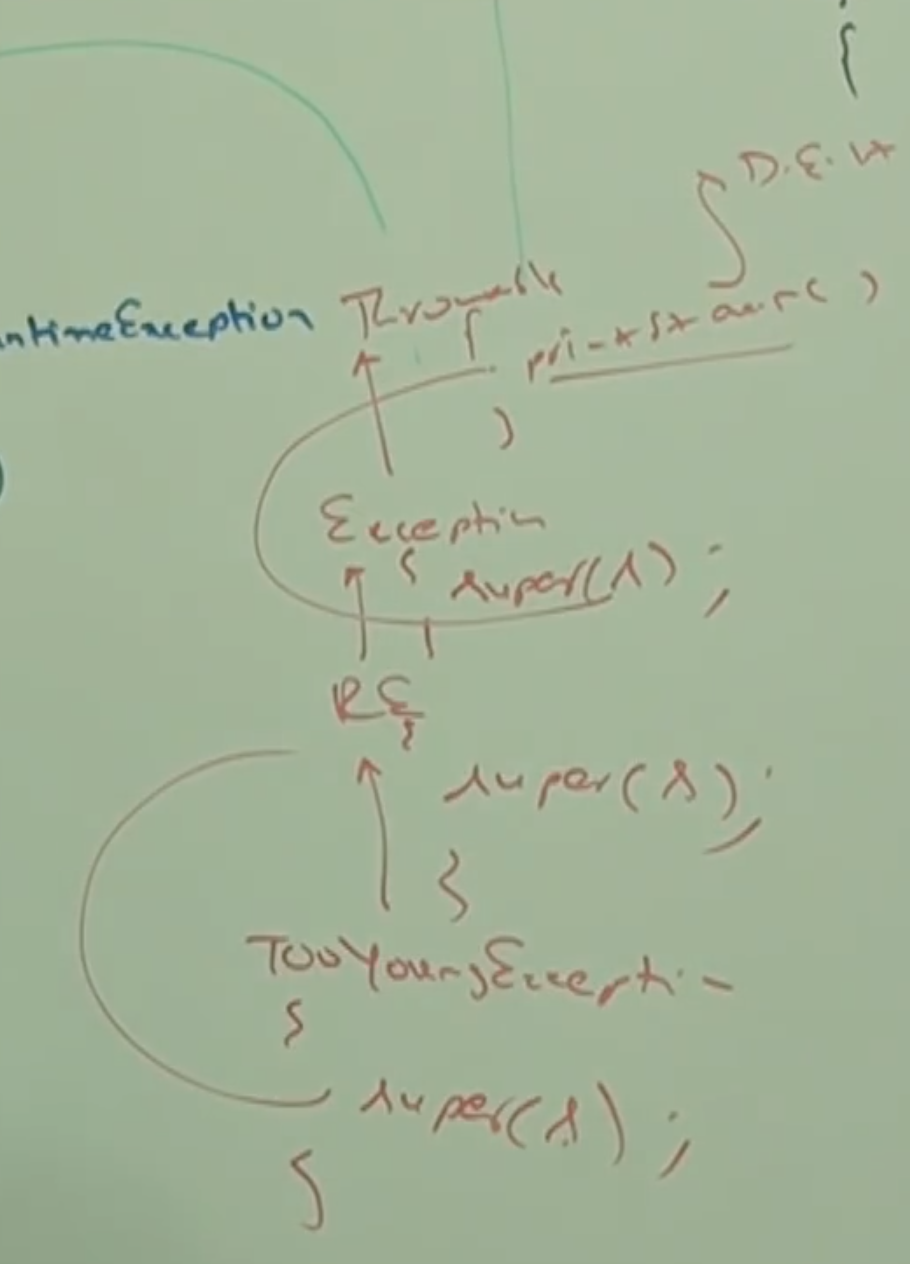
} else {

System.out.println(“You will get match details soon by email…!”);

}

}

}



Note:

1. Throw keyword is best suitable for user-defined or customized exceptions, but not for pre-defined exceptions.
2. It is highly recommended to define customized exceptions as unchecked. That is, we have to extend RuntimeException but not Exception.